Listing of Claims

Claims 1-24. (Canceled)

Claim 25. (Currently amended) A graphics system, comprising:

an input/output (I/O) bus;

a central processing unit (CPU) having an associated system memory, said CPU and said associated system memory coupled to said I/O bus, said CPU adapted to issue commands for rendering polygons of a graphical image;

a graphics accelerator ~~module~~ coupled to said ~~CPU and said associated system memory by a system~~ I/O bus, said graphics accelerator ~~module~~ comprising:

a cache for storing vertex data;

a cache controller configured to receive a command to render a polygon from said CPU, said cache controller checking said cache for previously cached vertex data for vertices of said polygon; and

said graphics accelerator ~~module~~ configured to utilize said vertex data to render pixel data for said polygon;

wherein said graphics accelerator caches vertex data received from said I/O bus to reduce the number of data transfers of vertex data across said I/O bus required to render polygons in response to said commands issued from said CPU.

Claim 26. (Previously presented) The graphics system of claim 25, further comprising: a state machine for directing said cache controller to update said cache.

Claim 27. (Previously presented) The graphics system of claim 25, wherein said CPU provides an index value for each vertex of a polygon to be rendered and said cache controller checks said cache for entries having said index value.

Claim 28. (Previously presented) The graphics system of claim 25, wherein said cache controller requests a transfer across said system bus from said system memory of any additional vertex data not present in said cache which is required to render said polygon.

Claim 29. (Previously presented) The graphics system of claim 28, wherein vertex data transferred into said graphics module is written into said cache for use in rendering subsequent polygons.

Claim 30. (Currently amended) A graphics system, comprising:

~~a system bus~~ an input/output (I/O) bus;

a central processing unit (CPU) coupled to said ~~system~~ I/O bus, said CPU adapted to issue requests to render polygons of a graphical image;

a system memory coupled to said ~~system~~ I/O bus, said system memory including a transfer buffer for storing vertex data associated with vertices of polygons to be rendered; and

a graphics accelerator ~~module~~ coupled to said ~~system~~ I/O bus for rendering polygons, comprising:

a cache for storing vertex data;

a direct memory access engine for transferring vertex data from said transfer buffer to said cache;

a cache controller configured to receive a request to render a polygon from said CPU which includes index values of vertices of said polygon, said cache controller checking said cache for entries having said index values and obtaining any additional required vertex data by directing said direct memory access engine to transfer required vertex data from said transfer memory; and

said graphics accelerator ~~module~~ configured to utilize said vertex data to render pixel data for said polygon;

wherein said graphics accelerator caches vertex data received from said I/O bus to reduce the number of data transfers across said I/O bus required to render polygons in response to said requests issued from said CPU.

Claim 31. (Previously presented) The graphics system of claim 30, further comprising: a state machine for directing said cache controller to update said cache.

Claim 32. (Previously presented) The graphics system of claim 30, wherein said direct memory access engine writes transferred vertex data into said cache, whereby said cache is updated for use in rendering at least one subsequent polygon.

Claim 33. (Currently amended) The graphics system of claim 30, wherein a transfer of vertex information for a polygon requires a plurality of data transfers across said ~~system~~ I/O bus, whereby use of cached vertex data reduces the number of data accesses required for rendering a polygon.

Claim 34. (Currently amended) The graphics system of claim 30, wherein said CPU is coupled to said ~~system~~ I/O bus by a graphics bridge.

Claim 35. (Previously presented) The graphics system of claim 34, wherein said system memory is connected to said graphics bridge.

Claim 36. (Previously presented) A computer as in claim 30 in which said cache has a memory mapped storage space for the data associated with said vertices.

Claim 37. (Currently amended) In a graphics system having a CPU and associated system memory coupled to a graphics accelerator ~~module~~ by ~~a system~~ an I/O bus, a method of reducing data transfers across said ~~system~~ I/O bus to said graphics accelerator required to render polygons, comprising:

storing vertex data in a cache that is local to said graphics accelerator ~~module~~,

at said graphics accelerator ~~module~~, receiving a command from said CPU to render a polygon, said command identifying index values of vertices of said polygon;

said graphics accelerator ~~module~~ checking index values of said cache for vertex data of said vertices of said polygon;

said graphics accelerator ~~module~~ reading said cache to obtain vertex data for each vertex of said polygon having cached vertex data;

wherein said graphics accelerator caches vertex data received from said I/O bus to reduce the number of data transfers across said I/O bus required to render polygons in response to said command.

Claim 38. (Currently amended) The method of claim 37, further comprising:

for each vertex of said polygon not having cached vertex data, said graphics accelerator ~~module~~ performing a memory transfer operation to transfer required vertex data from said system memory.

Claim 39. (Previously presented) The method of claim 38 further comprising: rendering said polygon using vertex data for each of said vertices.

Claim 40. (Previously presented) The method of claim 37 further comprising: updating said cache with vertex data for vertices not having vertex data stored in said cache, wherein said updating includes:

creating an array of vertices in a memory,

indexing data for each of said vertices which is stored in said array,

selecting from said array vertices defining a polygon to be rendered, and

transferring to said cache said data for each of said selected vertices.

Claim 41. (Currently amended) In a graphics system having a CPU and associated system memory coupled to a graphics accelerator ~~module side~~ by ~~a system~~ an I/O bus, a method of reducing data transfers across said ~~system~~ I/O bus required to render polygons, comprising:

at said graphics accelerator ~~module~~, receiving a command from said CPU to render a polygon, said command identifying index values of vertices of said polygon;

said graphics accelerator ~~module~~ checking a cache for vertex data for said vertices of said

polygon, wherein said cache is local to said graphics accelerator ~~module~~;

for each vertex of said polygon having cached vertex data, said graphics accelerator ~~module~~ reading said cache to obtain vertex data;

for each vertex of said polygon not having cached vertex data, said graphics accelerator ~~module~~ performing a memory transfer operation to transfer required vertex data from said system memory;

wherein said graphics accelerator caches vertex data received from said I/O bus to reduce the number of data transfers across said I/O bus required to render polygons in response to commands from said CPU.

Claim 42. (Previously presented) The method of claim 41 further comprising: rendering said polygon using vertex data for each of said vertices.

Claim 43. (Currently amended) The method of claim 41 further comprising:

said graphics accelerator ~~module~~ updating said cache with transferred vertex data from said memory transfer operation.

Claim 44. (Previously presented) The method of claim 43, further comprising:

creating an array of vertices in a memory,

indexing data for each of said vertices which is stored in said array,

selecting from said array vertices defining a polygon to be rendered, and

transferring to said cache said data for each of said selected vertices.